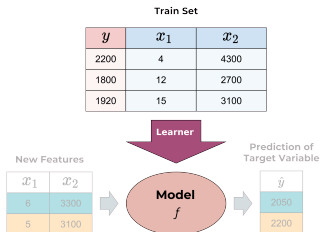


ML-Basics Learner



- Understand that a supervised learner fits models automatically from training data

-

SUPERVISED LEARNING EXAMPLE

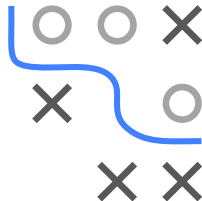
Imagine we want to investigate how working conditions affect productivity of employees.

- It is a **regression** task since the target *productivity* is continuous.
- We collect data about worked minutes per week (*productivity*), how many people work in the same office as the employee in question, and the employee's salary.

Features x		Target y
People in Office (Feature 1) x_1	Salary (Feature 2) x_2	Worked Minutes Week (Target Variable)
4	4300 €	2220
12	2700 €	1800
5	3100 €	1920

Diagram illustrating the dataset structure:

- The dataset consists of $n = 3$ samples (rows).
- Each sample has $p = 2$ features (x_1 and x_2).
- The target variable is y .
- The features are labeled $x_1^{(2)}$ and $x_2^{(1)}$.
- The target variable is labeled $y^{(3)}$.



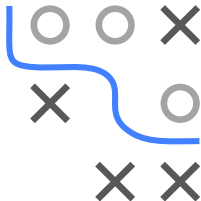
SUPERVISED LEARNING EXAMPLE / 2

How could we construct a model from these data?

We could investigate the data manually and come up with a simple, hand-crafted rule such as:

- The baseline productivity of an employee with salary 3000 and 7 people in the office is 1850 minutes
- A decrease of 1 person in the office increases productivity by 30
- An increase of the salary by 100 increases productivity by 10

=> Obviously, this is neither feasible nor leads to a good model



LEARNER DEFINITION

- The algorithm for finding our f is called **learner**. It is also called **learning algorithm** or **inducer**.
- We prescribe a certain hypothesis space, the learner is our means of picking the best element from that space for our data set.
- Formally, it maps training data $\mathcal{D} \in \mathbb{D}$ (plus a vector of **hyperparameter** control settings $\lambda \in \Lambda$) to a model:

$$\mathcal{I} : \mathbb{D} \times \Lambda \rightarrow \mathcal{H}$$

